SYSTEM AND METHOD FOR PROVIDING WEB BROWSER-BASED SECURE REMOTE NETWORK APPLIANCE CONFIGURATION IN A DISTRIBUTED COMPUTING ENVIRONMENT

Cross-Reference to Related Applications

This patent application is a conversion of U.S. provisional patent applications, Serial No. 60/309,835, filed August 3, 2001, pending; and Serial No. 60/309,858, filed August 3, 2001, pending; the priority dates of which are claimed and the disclosures of which are incorporated by reference.

Field of the Invention

The present invention relates in general to secure network appliance configuration and, in particular, to a system and method for providing Web browser-based secure remote network appliance configuration in a distributed computing environment.

Background of the Invention

Enterprise computing environments generally include both localized intranetworks of interconnected computer systems and resources internal to an organization and geographically distributed internetworks, including the Internet. Intranetworks make legacy databases and information resources available for controlled access and data exchange. Internetworks enable internal users to access remote data repositories and computational resources and allow outside users to access select internal resources for completing limited transactions or data transfer.

Increasingly, network appliances, or simply "appliances," are being deployed within intranetworks to compliment and extend the types of services offered. As a class, network appliances have closed architectures and often lack a standard user interface. These devices provide specialized services, such as

electronic mail (email) anti-virus scanning, content filtering, file, Web and print service, and packet routing functions.

Ideally, network appliances should be minimal configuration devices, which are purchased, plugged into a network, and put into use with no further modification or change. Analogous to a cellular telephone, a network appliance should ideally provide the service promised without requiring involved configuration and setup by individual users or administrators.

Nevertheless, configuring newly-installed appliances remains a complicated and confusing endeavor. Appliance configuration is generally vendor-specific and device-dependent. The lack of a user interface allows only indirect configuration and setup. Configuration often takes several steps. From a physical connectivity standpoint, appliance configuration typically requires operating a manual control panel, reconfiguring an installed appliance from a factory set of default settings or performing a myriad of other device-dependent operations to affect a configured setup. Consequently, a higher than average level of user sophistication is required to avoid a confusing, incorrect or potentially catastrophic outcome.

In addition, operational software and firmware must also be properly configured as part of an initial setup. Often, a full software suite, including operating system, must be installed prior to initializing the appliance. In addition, the network protocol stack must be configured to operate within the specific installed network topology into which the device is deployed.

Finally, various policies must be installed and operationally enforced on each appliance. Appliances offering plug-and-play installation generally lack the default settings necessary to enforce security and administrative policies. As well, until fully configured, these devices enjoy potentially free rein over a network domain and pose a serious security risk to an entire enterprise.

For instance, replay attacks are possible during device configuration. A configuration packet could be intercepted by a hostile agent and later re-sent ("replayed") with altered settings to reset the configuration and create a security breach.

In addition to per-device configuration and setup considerations, the deployment of appliances can create network management concerns. For instance, a large population of deployed appliances can drastically increase network management workload. Vendor-specific and device-dependent settings necessitate individualized attention to each successive appliance installation. A rich network environment having a multitude of heterogeneous systems and appliances can quickly overwhelm a network administrator and make the task of identifying unconfigured devices difficult and time consuming.

In the prior art, the dynamic host configuration protocol (DHCP) provides a partial solution. DHCP allows a TCP/IP-compatible device to be dynamically assigned a network address within a pre-defined network domain. A DHCP server maintains a table of the network addresses assigned to each interconnected device, thereby preventing address conflicts. Network address assignments are "pushed" to each newly-connected device. However, DHCP servers are limited to configuring network addresses and fail to provide policy and device parameter configuration and setup.

Therefore, there is a need for an approach to providing remote secure configuration of network appliances from a standardized user interface. Preferably, such an approach would offer a Web browser-based solution allowing configuration from a ubiquitous and widely available interfacing means. Such an approach would further provide a standardized interface for appliance configuration and setup in a vendor-neutral and device-independent fashion.

There is a further need for an approach to providing automatic configuration of network appliances during initialization upon deployment into a network domain. Preferably, such an approach would provide a complete bootstrap solution with minimal user interaction. Furthermore, such an approach would preferably realize a cellular telephone service model of purchase, plug in and use.

There is a further need for an approach to providing network-based configuration of network appliances that substantially minimizes the potential for creating security risks and, in particular, preventing replay attacks.

Summary of the Invention

The present invention provides a system and method for remotely configuring a network appliance deployed within a network domain. A configuration client executes a Web browser upon which is loaded an applet for performing remote appliance configuration. The applet is initially retrieved from a centralized network operations center, which maintains a set of applets customized for each separate network domain and individual configurations for various network appliances. The configuration client, via the applet, broadcasts a "ping" query message to all appliances and receives back from each a response indicating a configuration state. An appliance configuration for each unconfigured network appliance is requested from the network operations center. The network operations center returns configuration parameters to the configuration client and a configuration packet is sent to each unconfigured appliance. Upon the successful configuration of each appliance, the configuration client instructs the appliance to begin a remote management session. Otherwise, the configuration packet is resent or the configuration client waits for the installation to complete.

An embodiment of the present invention provides a system and a method for providing Web browser-based remote network appliance configuration in a distributed computing environment. A query message is broadcast from an applet executing within a Web browser to one or more network appliances. The network appliances are interconnected within a bounded network domain defined by a common network address space. A response message containing network settings, including a physical network address, is received by the applet from at least one such network appliance responsive to the query message and processed. A configuration packet is generated and sent using the physical network address for each at least one such network appliance sending a response message and requiring configuration.

A further embodiment provides a system and method for remotely configuring a network appliance deployed within a distributed computing environment. A response message containing network settings is sent from at

least one network appliance responsive to a query message broadcast over a specified network domain within which the at least one network appliance operates. A configuration package for the at least one network appliance is generated. The configuration package contains centrally managed network settings customized for the at least one network appliance. The configuration package is installed on the at least one network appliance as part of an initialization bootstrap operation.

Still other embodiments of the present invention will become readily apparent to those skilled in the art from the following detailed description, wherein is described embodiments of the invention by way of illustrating the best mode contemplated for carrying out the invention. As will be realized, the invention is capable of other and different embodiments and its several details are capable of modifications in various obvious respects, all without departing from the spirit and the scope of the present invention. Accordingly, the drawings and detailed description are to be regarded as illustrative in nature and not as restrictive.

Brief Description of the Drawings

FIGURE 1 is a block diagram showing a system for providing Web browser-based secure remote network appliance configuration in a distributed computing environment.

FIGURE 2 is a block diagram showing the software modules of the network operations center of FIGURE 1.

FIGURE 3 is a block diagram showing the software modules of the configuration client of FIGURE 1.

FIGURE 4 is a block diagram showing the software modules of an exemplary network appliance of FIGURE 1.

FIGURE 5 is a process flow diagram showing a remote network appliance configuration, as performed by the system of FIGURE 1.

FIGURE 6 is a data structure diagram showing a configuration packet served by the configuration client of FIGURE 1.

FIGURE 7 is a flow diagram showing a method for providing Web browser-based secure remote network appliance configuration in a distributed computing environment, in accordance with the present invention.

FIGURE 8 is a flow diagram showing the process performed by the network operations center of FIGURE 2.

FIGURES 9A and 9B are flow diagrams showing the process performed by the configuration client of FIGURE 1.

FIGURE 10 is a flow diagram showing the process performed by the network appliance of FIGURE 4.

Detailed Description

FIGURE 1 is a network diagram 10 showing a system for providing Web browser-based secure remote network appliance configuration in a distributed computing environment, in accordance with the present invention. The distributed computing environment is preferably TCP/IP compliant. A plurality of individual network appliances (or simply "appliances") 11a-c are interconnected via an intranetwork 13. Each of the appliances 11a-c is autonomously configured and provides specified functionality, such as electronic mail (email) anti-virus scanning, content filtering, packet routing, or file, Web, or print service. Other forms of appliance services are feasible, as would be recognized by one skilled in the art.

In addition to providing the specified functionality, the various appliances 11a-c are autonomously self-configured and self-managed, as further described below beginning with reference to FIGURE 4. The appliances 11a-c are remotely configured through a configuration client 16 executing within a bounded network domain defined by a common network address space. The configuration client 16 includes a Web browser 17 upon which an applet 23 executes to transparently install and configure each of the interconnected appliances 11a-c.

Upon the physical connection of each new appliance 11a-c onto the intranetwork 13, an administrator executes a configuration application on the configuration client 16 via the Web browser 17. The Web browser 17 provides a user-friendly and standardized user interface for configuring appliances 11a-c in a

device-independent and vendor-neutral manner. The configuration application executes the applet 2, which broadcasts a "ping" query message to all appliances 11a-c on the intranetwork 13. In response, each appliance 11a-c sends a response back to the configuration client 16, which then determines those appliances 11a-c requiring configuration and setup.

For each unconfigured appliance 11a-c, the configuration client 16 requests configuration parameters from a centralized network operations center (NOC) 12 in a secure session.

The network operations center 12 determines the parameters necessary to properly configure the unconfigured appliance 11a-c in accordance with applicable security and administration policies. The configuration parameters are sent to the requesting configuration client 16. Upon receiving the set of configuration parameters for each new appliance 11a-c, the configuration client 16 generates a configuration packet, which is customized for and sent to each unconfigured appliance 11a-c. Upon the successful installation of each configuration packet by the appliances 11a-c, the configuration client 16 sends a "kick-start" packet to initiate a secure remote management session on each appliance 11a-c, such as described in commonly-assigned related U.S. patent application Serial No. _____, entitled "System And Method For Providing A Framework For Network Appliance Management In A Distributed Computing Environment," filed January 25, 2002, pending, the disclosure of which is incorporated by reference.

The appliance configuration performed by the configuration client 16 is system independent and can be executed by any client interconnected within the same network domain as the appliances being configured. Accordingly, each new configuration client 16 initially requests an applet from an applet server 15 executing on the network operations center 12 via a secure session. The applet server 15 is coupled to an applet database 14 to allow customization of the configuration functions performed within each individual network domain. Upon receipt of the applet, the configuration client 16 can proceed to configure the individual appliances 11a-c.

Each appliance 11a-c is interconnected via an intranetwork 13 which is, in turn, interconnected to an internetwork 20, including the Internet, via a firewall 21 and border router 22. The configuration client 16 is also interconnected via the intranetwork 13 and shares the same network domain with the appliances 11a-c. The network operations center 12 is external to the intranetwork 13 and is only accessible as a remote host via the internetwork 20. Accordingly, the configuration parameter and applet request functions are transacted with each appliance 11a-c in a secure session, preferably using the Secure Hypertext Transport Protocol (HTTPS). Other network configurations, topologies and arrangements of clients and servers are possible, as would be recognized by one skilled in the art.

The individual computer systems, including servers and clients, are general purpose, programmed digital computing devices consisting of a central processing unit (CPU), random access memory (RAM), non-volatile secondary storage, such as a hard drive or CD ROM drive, network interfaces, and peripheral devices, including user interfacing means, such as a keyboard and display. Program code, including software programs and data, are loaded into the RAM for execution and processing by the CPU and results are generated for display, output, transmittal, or storage.

FIGURE 2 is a block diagram showing the software modules 30 of the network of FIGURE 1. The network operations center 12 includes three modules: status monitor 31, status daemon 32 and applet server 15. The applet server 15 executes as part of the network operations center 12. The status monitor 31 receives periodic status reports from the individual network appliances 11a-c (shown in FIGURE 1). Each status report is recorded and registered in an appliance status table 33, which notes the appliance user identifier (UID) and time of each report. The status daemon 32 executes as an independent process that periodically awakens and examines the appliance status table 33 to determine whether any of the appliances 11a-c have failed to report. As necessary, an alert is generated to inform an administrator of a potentially faulty appliance.

The applet server 15 includes three modules: applet engine 34, database 35, and crypto 36. The applet engine 34 downloads individual applets 23 maintained in the applet database 14 to requesting configuration clients 16 (shown in FIGURE 1) via a secure session. A library of applets 37 are maintained to allow customization of the various configuration applications executing within the Web browsers 17 each configuration client 16.

The database module 35 interfaces to the applet database 14 to access the applets 37 maintained therein. In the described embodiment, the applet database 14 is a structured query language (SQL) based database. The applets 37 are stored as structured records indexed by client identifiers.

The crypto module 36 provides asymmetric (public key) and symmetric encryption. Both forms of cryptography are needed to transact a secure session with each appliance 11a-c. As well, the network operations center 12 uses the crypto module 36 to digitally sign and encrypt the applets 37.

The network operations center 12 includes a message queue 38 through which instructions to the applets 23 (shown in FIGURE 1) deployed on the individual configuration clients 16 are communicated. The configuration clients 16 execute in an event-driven manner. Periodically, each configuration client 16 checks the message queue 38 for new instructions which are transparently executed by the applet 23.

In the described embodiment, five types of messages are communicated between the network operations center 12 and the configuration clients 16, as follows:

sendRefresh(): Sends a message to message queue 38 instructing the applet 23 to refresh the list of appliances 11a-c that are on the network.

SendKick(): Sends a message to message queue 38 instructing the applet 23 to send out a kick-start packet to the appliance 11a-c with the given media access controller (MAC) address.

Parameters:

MAC: The MAC address of the appliance 11a-c to which the kick-start packet will be sent. Should be in "AA:BB:CC:00:11:22" format.

sendConfig(): Sends a message to message queue 38 instructing the applet 23 to send a "CONFIG" configuration packet to the appliance 11a-c with the given MAC address.

Parameters:

MAC: The MAC address of the appliance 11a-c to which the configuration packet will be sent. Should be in "AA:BB:CC:00:11:22" format.

Hostname: Value to be assigned as the hostname.

Domain: Value to be assigned as the domain name.

IP: Value to be assigned as the IP address.

Netmask: Value to be assigned as the network mask.

Gateway: Value to be assigned as the internet gateway.

DNS1: Primary domain-name server.

DNS2: Secondary domain-name server.

String getList(): Returns a list of select appliances 11a-c with current network configuration in an internal appliance list in a configuration client 16.

Parameters:

Filter: Value that determines which appliances 11a-c are returned. If the value is "0," all appliances are returned; and if the value is "2," only configured appliances are returned

Return Value:

The return value is a String that contains the select appliances, and current configuration information. The return value is a pipe-symbol ("|") delimited for every network parameter. An example return value is:

00:B0:D0:11:22:33:test1,mycio.com,127.0.0.1,255.255.255.128,0.0.0.0,0.0.0.0,0.0.0.0|00:11:22:33:44:55:test2,mycio.com,127.0.0.1,255.255.255.128,0.0.0.0,0.0.0.0

getStatus(): Returns the status of the *sendConfig* message. Returns "0" if no SUCCESS or FAILED packet has yet been received from an appliance 11a-c; "1" if a SUCCESS packet was received; and "–1" if a FAILED packet was received.

FIGURE 3 is a block diagram showing the software modules 40 of a configuration client 16 of FIGURE 1. The configuration client 16 includes a Web browser 17 executing an applet 23. In the described embodiment, the Web browser 17 is a HTML-compatible Web browser, such as the Internet Explorer, licensed by Microsoft Corporation, Redmond, Washington, capable of executing downloadable programs, including applets, written in an interpretable programming language, such as the Java programming language.

The applet includes three functional modules: status 41, configuration and packet generation 42, and completion 43. The status module 41 broadcasts a query message to the interconnected network appliances 11a-c (shown in FIGURE 1) and processes response messages received back to determine the configuration of each appliance 11a-c. The status of each appliance 11a-c is maintained in a configured appliances list 44. The configuration and packet generation module 42 receive configuration parameters from the network operations center 12 (shown in FIGURE 1) and generates a configuration packet for downloading to an unconfigured appliance 11a-c. The completion module 43 receives a status message from each unconfigured appliance 11a-c indicating whether the configuration packet was successfully installed. A configuration packet will be re-sent to any appliance 11a-c that fails to successfully complete configuration.

FIGURE 4 is a block diagram showing the software modules 50 of an exemplary network appliance 11a of FIGURE 1. Application-specific logic has been omitted for clarity. As pertains to autonomous configuration and management, each network appliance 11a includes four modules: bootstrap module 51, crypto 52, installer 53, and status daemon 54. The bootstrap module 51 executes upon the initial installation of the appliance 11a onto the intranetwork 13. The bootstrap module 51 sends a response message in reply to a broadcasted

"ping" query message from the configuration client 16. The response message includes the current configured network settings in use by the appliance 11a. For an unconfigured appliance 11a, the response packet includes only the media access controller (MAC) address used by the appliance 11a. As well, the bootstrap module 51 sends a response message to any subsequent query messages sent by the configuration client 16 and includes all currently in-use configured network settings, as maintained in the appliance configuration 55.

Upon receiving a configuration packet from the configuration client 16, the bootstrap module 51 installs and sets up the various software applications to be executed by the appliance 11a. The software can include the operating system and any application-specific logic integral to providing the service performed by the appliance 11a. Through the use of the network operations center 12 and configuration client 16, the appliance 11a can be configured and managed remotely and in compliance with applicable security and administrative policies. Accordingly, the autonomous configuration and self-management of each network appliance 11a-c can enable a vendor to provide a complete service model, whereby installations are handled autonomously and without significant end-user intervention.

The crypto module 52 provides asymmetric (public key) and symmetric encryption. Both forms of cryptography are needed to transact a secure session with the network operations center 12 and a component server (not shown) used to manage and update the suite of applications 56 installed on the appliance 11a. The installer 53 installs applications received from a component server. Finally, the status daemon 54 periodically awakens and sends a report of the health and status of the network appliance 11a to the network operations center 12. The status report identifies the reporting appliance 11a and provides machine-specific data, including the load on the processor, available disk space and application-specific information, such as the number of emails passing through the device. The status report is referred to as a "SecureBeat."

Each software module of the network operations center 12, configuration client 16 and exemplary appliance 11a is a computer program, procedure or

module written as source code in a conventional programming language, such as the C++ programming language, and is presented for execution by the CPU as object or byte code, as is known in the art. The various implementations of the source code and object and byte codes can be held on a computer-readable storage medium or embodied on a transmission medium in a carrier wave. The network operations center 12, configuration client 16 and exemplary appliance 11a operate in accordance with a sequence of process steps, as further described beginning below with reference to FIGURE 7.

FIGURE 5 is a process flow diagram showing a remote network appliance configuration, as performed by the system of FIGURE 1. Each network appliance is autonomously configured by a configuration ("config") client 61. Upon the installation of a new appliance on the intranetwork 13 (shown in FIGURE 1), or as necessary to ascertain the current appliance configuration, the configuration client 61 broadcasts a "ping" query message (step 65) to all appliances 62 currently interconnected within the bounded network domain. In response, each appliance 62 sends a response message (step 66) back to the configuration client 61. Each response includes the current configured network settings in use by each appliance 62. A response containing only the media access controller (MAC) address of the appliance 62 indicates that the appliance is currently unconfigured.

For each of the unconfigured appliances, the configuration client 61 sends a configuration packet request message (step 67) to the network operations center 63 via a secure session. The network operations center 63 determines the correct configuration settings required by the appliance to be configured by referencing an appliance status table 33 (shown in FIGURE 2). The network operations center 63 generates a set of configuration parameters, which are sent (step 68) back to the requesting configuration client 61. The secure session is closed and the configuration client 61 forms a configuration packet for the unconfigured appliance 62.

The configuration client 61 sends the configuration packet (step 69) to the unconfigured appliance 64 where the configuration packet is processed and installed. The appliance 64 sends a "SUCCESS" message (step 70) to the

configuration client 61 upon the successful configuration of the appliance. In response, the configuration client 61 returns a kick-start message (step 71) back to the appliance 64 to initiate an autonomous SecureBeat management session. Thereafter, the ongoing management of the appliance 64 is remotely facilitated by the network operations center 63.

If the configuration is unsuccessful, the appliance 64 sends a "FAILURE" message (step 72) back to the configuration client 61, which resends the configuration packet (step 69) until successful.

If the appliance 64 is still in the process of configuring, the appliance 64 sends an unconfigured message (step 73) back to the configuration client 61, which then waits until the appliance 64 has been configured. Thereafter, a SecureBeat management session is initiated.

FIGURE 6 is a data structure diagram showing a configuration packet 80 served by the configuration client 16 of FIGURE 1. Each configuration packet 80 contains the parameters described above with reference to FIGURE 2.

While not necessary to completing an initial appliance configuration, the primary and secondary domain name server parameters 78 and 88, respectively, are optional and are provided for network administrative convenience.

FIGURE 7 is a flow diagram 100 showing a method for providing Web browser-based secure remote network appliance configuration in a distributed computing environment, in accordance with the present invention. The individual components, including network operations center 12, configuration client 16 and individual network appliances 11a-c, execute independently. Each of the components must be initialized and started (blocks 101-103) prior to appliance configuration. Upon respective initialization and starting, each component proceeds independently, as further described below with reference to FIGURES 8-10.

FIGURE 8 is a flow diagram 110 showing the process performed by the network operations center 12 of FIGURE 2. Network operations center 12 begins by connecting to a configuration client 16 (shown in FIGURE 1) requesting an applet 23 (shown in FIGURE 2) (block 111). An applet 23 is downloaded to the

configuration client 16 (block 112). Each configuration client 16 executes the applet 23 in a Web browser 17.

Following applet download (blocks 111-112), the network operations center 12 executes an iterative processing loop (blocks 113-119). During each iteration (block 113), a secure session is established with a configuration client 16 (block 114). Upon establishing a secure session, a configuration packet request is received (block 115). The network operations center 12 looks up the configuration 40 (shown in FIGURE 2) for the configured appliance 11a and generates configuration parameters (block 116). The configuration parameters are downloaded to the configuration client 16 (block 117), after which the secure session is closed (block 118). Processing continues (block 119) until the process is terminated or halted.

FIGURES 9A and 9B are flow diagrams 120 showing the process performed by the configuration client 16 of FIGURE 1. The configuration client 16 begins by broadcasting a "ping" query message to all network appliances 11a-c (block 121) interconnected within the bounded network domain. The configuration client 16 then executes an iterative processing loop (blocks 122-133) for each appliance 11a-c.

During each iteration (block 122), a response from an appliance 11a is received (block 123) and processed as follows. If the response from the appliance 11a indicates that the appliance is not presently configured (block 124), a configuration parameters request is sent to the network operations center 12 (block 125). The network operations center 12 generates a set of configuration parameters which are then received (block 126) and formed into a configuration packet for the unconfigured appliance 11a (block 127). The configuration packet is sent to the appliance 11a (block 128).

The configuration client 16 awaits a status response from the appliance 11a (block 129). If the configuration succeeds (block 130), the configuration client 16 sends a kick-start packet to the appliance 11a (block 131), instructing the now-configured appliance 11a to initiate an autonomous SecureBeat management. Otherwise, if the configuration is not successful (block 130) and has failed (block

132), the configuration packet is sent again to the appliance 11a (block 128). Otherwise, the configuration client 16 waits for the completion of configuration by the appliance 11a (block 133), after which a kick-start packet is sent to the appliance 11a (block 131). Processing continues (block 134) until the process is terminated.

FIGURE 10 is a flow diagram 140 showing the process performed by the network appliance 11a of FIGURE 4. Shortly following deployment into a network domain, or as necessary, a "ping" query message is received from a configuration client 16 (shown in FIGURE 1) (block 141). In response to the query message, the network appliance 11a generates and sends a response back to the requesting configuration client 16 (block 142).

The response message includes the current network setting and configuration 55 (shown in FIGURE 4) used by the network appliance 11a. If the network appliance is not currently configured (block 143), a configuration packet is received from the configuration client 16 (block 144) and installed (block 145). If the installation is successful (block 145), a "success" response message is sent back to the configuration client 16 (block 146). The network appliance 11a then receives a kick-start packet from the configuration client 16 (block 147) instructing the network appliance 11 a to initiate a remote SecureBeat management session (block 148). If installation is not successful (block 145) and fails (block 149), a "failure" response is sent back to the configuration client 16 (block 150), after which a further configuration packet is received from the configuration client 16 (block 144). Otherwise, if installation is still being performed (block 149), an "unconfigured" response is sent to the configuration client 16 (block 151) and the network appliance waits for configuration completion (block 152), after which a kick-start packet is received (block 147) and remote SecureBeat management session initiated (block 148).

While the invention has been particularly shown and described as referenced to the embodiments thereof, those skilled in the art will understand that the foregoing and other changes in form and detail may be made therein without departing from the spirit and scope of the invention.